Exploring Structure and Texture in Clothing Embeddings

Anelise Newman
18.065 Final Project
Massachusetts Institute of Technology
apnewman@mit.edu

Abstract

The structure of a garment–the silhouette, the decorative details, how the seams fit together–is integral to the style and the construction of a piece of clothing. Work in deep learning for fashion focuses disproportionately on textural attributes of clothing, to the detriment of other design details. The goal of this project is to use a neural network to learn a texture-invariant, structure-sensitive embedding for images of clothing. To do so, I gather a new dataset consisting of both photographs and detailed design sketches and train a network to map both types of data to a shared embedding space. The resulting embedding is less sensitive to color and more sensitive to design details than one trained on photographs alone.

1. Introduction

There is a disconnect between how computer vision algorithms understand clothing and how designers and consumers of clothing do. In particular, a common technique in computer vision is to decompose garments into their "shape" (a 1-d segmentation mask indicating the border of the garment) and "texture" (a color pattern used to color in the segmentation) [12, 14, 19]. This representation obscures more specific design features that are important to garment construction and fashionability: for example, the type of collar on a shirt, the fullness of a skirt, or the style of the sleeves.

In this project, I make progress towards creating a clothing embedding that captures these details of garment structure. I show that an embedding technique based on Siamese networks and contrastive loss, which has previously worked for non-deformable items like furniture, can also give reasonable results for clothing. I experiment with different image preprocessing techniques to improve the extent to which the model generalizes to different textures. Furthermore, I propose using a new source of data–paired photographs and design sketches from sewing patterns–to create a texture-invariant embedding that captures design elements. I collect the Simplicity patterns dataset and use it to train a joint embedding for natural images and line drawings. The resulting embedding is able to detect subtle design features across articles of clothing, and with more data could lead to precise structural embeddings for clothing[1].

2. Related work

2.1. Learning embeddings

Neural networks excel at reducing a large, high-dimensional feature space into a compact subspace, or embedding, that efficiently maps the high-level characteristics of the input data. A network that compresses a high-dimensional stimulus into a compact feature vector is called an encoder. For example, an image encoder can convert a natural image into a feature vector that is convenient for tasks such as image classification or image captioning, while a text encoder might take in a complex document and produce an embedding that is useful for document classification or next-word prediction.

There are many ways to learn a useful embedding. Supervised learning approaches train a network to perform a specific task for which we have ample ground-truth human data–for example, image classification on ImageNet [3], or caption prediction on the COCO dataset [23, 26]. Even though they are trained for a specific task, the embeddings generated by these encoders often generalize well to other tasks, which is why ImageNet pretraining is such a common practice in computer vision. More recently, unsupervised learning approaches have attempted to learn useful embeddings *without* using any explicit human annotations, by modeling and predicting patterns inherent in raw data. Unsupervised approaches (including, most recently, contrastive approaches), have had success in computer vision [2, 20, 24] and huge success in Natural Language Processing [4, 16, 22].

A middle-ground between these two approaches are techniques that learn a similarity metric by training a a net-

[1]For code and additional results, please see: https://github.com/a-newman/clothing-embedding

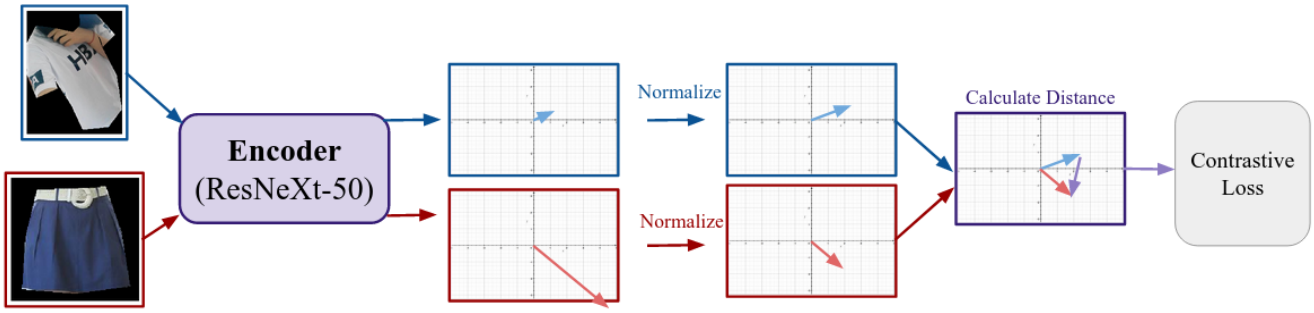

Figure 1. Architecture of the appearance-based clothing embedding model. A pair of images is passed through the same encoder. After normalization, the feature vectors are passed through a contrastive loss and the loss is backpropagated using stochastic gradient descent. The contrastive loss pushes the vector representations for the two images closer together if they come from a matching pair and farther apart if they do not.

work to distinguish between same-different pairs. I will refer to these techniques as "semi-supervised" embeddings. In natural language processing, the popular Word2Vec word embedding [15] attempts to predict the current word based on a window of context words. In practice, the model predicts the log probability of the current word given the context and uses negative sampling to enforce that this probability is higher for the true current word than for a randomly sampled word. In computer vision, Siamese networks [13] use a contrastive loss to learn an embedding space where matching data points are close together and non-matching data points are far apart. This project was heavily inspired by the work of Bell et al., which used a Siamese network to learn an embedding for pictures of furniture [1]. These semi-supervised similarity embeddings are attractive for fashion applications because they can make use of large fashion image datasets that often come with paired images of the same item.

2.2. Deep learning for fashion

Fashion embeddings focus on images of garments, outfits, or descriptions thereof. Previous work has focused on generating fashion embeddings to measure and evaluate fashionability [11, 21], or detect the main product in an image [27]. More recently, interest has grown in generating new images of clothing, either by "warping" clothing from existing product images onto an image of a person or by generating new clothes entirely from scratch [17, 19, 8, 5, 7, 12, 14, 9]. A common approach to generative models of clothing is to first generate the shape of the garment in the form of a segmentation map and then "fill in" the texture/color components [12, 14]. However, modeling a piece of clothing as essentially a flat canvas that can be filled in with a consistent texture glosses over much of the rich internal structure of a garment. In this work, I will seek to recover some of that complexity in fashion embeddings.

3. An appearance-based clothing embedding

In this section, I train a Siamese network to produce an embedding for articles of clothing using the DeepFashion2 dataset[2] [6]. I experiment with different input formats for the data to see how they influence the ability of the model to identify textural versus structural similarity of garments.

3.1. Model

I use a Siamese model like that introduced in [13]. The encoder architecture is an ImageNet-pretrained ResNeXt-50 [25]. At each iteration, the model receives a minibatch consisting of multiple pairs of samples. For each pair, both elements are passed in sequence through the same encoder with the same weights, producing two output features of size 256. The feature vectors are then normalized to unit length (as in [1]). Finally, the two feature vectors are passed to a contrastive loss, defined as:

$$\mathcal{L}_c = \frac{1}{2}[(y)(d)^2 + (1-y)(max(0, m-d))^2]$$

where y is an indicator variable that is set to 1 if the two inputs match and 0 otherwise, d is the euclidian distance between the two output feature vectors, and m is a margin that specifies the minimum distance that should exist between two non-matching inputs. Intuitively, this loss encourages the feature vectors for a matching pair to be as close together as possible, while the vectors for a non-matching pair should be a distance of at least m apart. Here, we set $m = \sqrt{0.2}$ as in [1]. This model is visualized in Figure 1.

3.2. Dataset

I train this model using the DeepFashion2 dataset [6]. DeepFashion2 is one of the biggest available datasets for

[2] https://github.com/switchablenorms/DeepFashion2

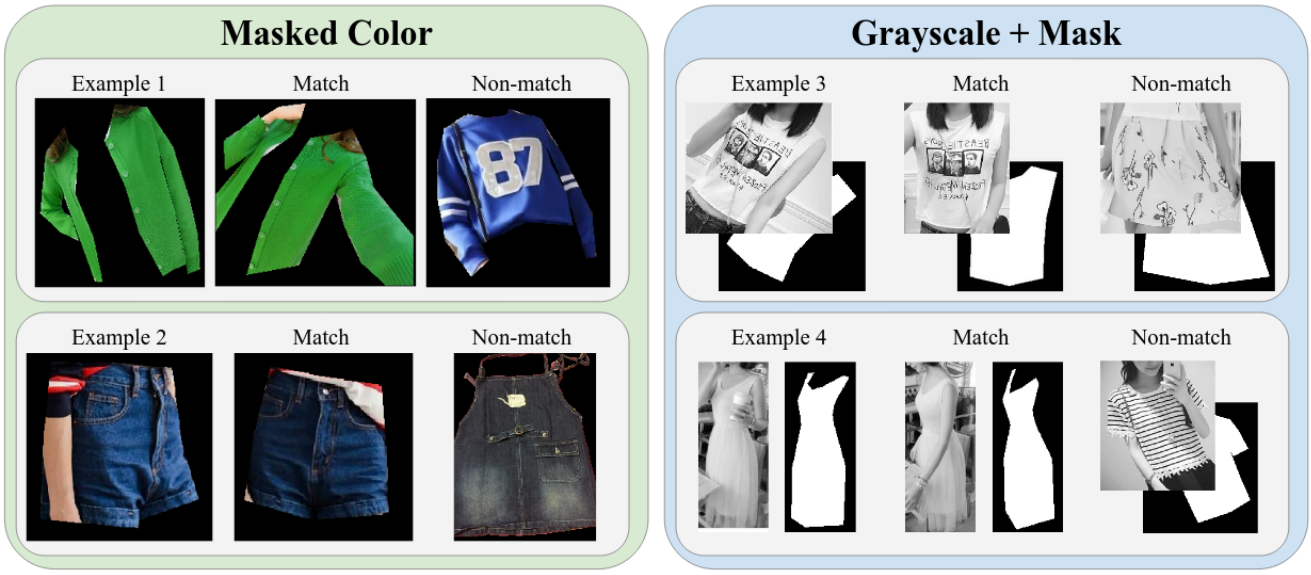

Figure 2. **Example contrastive pairs from the DeepFashion2 dataset.** Before being fed to the model, each image is cropped to the bounding box of the garment. Matching pairs are alternate views of the same garment, while non-matching pairs show different garments. I experiment with two methods of preprocessing. For the **masked color** method (left), background pixels are masked out. For the **grayscale + mask** method (right), the entire grayscale image is fed in along with a binary mask in a second channel.

fashion-related tasks. The dataset contains 491,000 images containing 801,000 clothing instances. This is split into 391,000 training images, 34,000 validation images and 67,000 test images. It also contains many useful per-item annotations, including garment category labels, bounding boxes, per-pixel segmentations, and item ids. In this project, I use the garment item ids to construct the matching and non-matching pairs used during training. I use the bounding boxes and segmentation masks to normalize the data before passing it into the network in order to improve performance.

3.2.1 Forming image pairs.

The contrastive technique for generating an embedding requires that we be able to construct paired sets of images. Fortunately, for many of the clothing items in the dataset, DeepFashion2 contains multiple photographs of the same garment, often in different poses or different contexts. As a preprocessing step, I filtered the dataset down to only garments that had at least two images in the dataset. This produced 36,657 unique garments with a total of 308,239 instances in the training set and 6,044 garments with 51,964 instances in the validation set. I will refer to this subset of the dataset as DeepFashion2-Duplicates.

3.2.2 Contrastive data loading.

During training, I iterate through each clothing instance in the DeepFashion2-Duplicates data set (in a random order). I then sample a second image to complete the image pair. With equal probability, I sample a matching or non-matching second image. A matching image is another photograph of the same clothing item. A non-matching image is a randomly selected image of another item. This results in over 9.5 billion possible image pairings that the model could see during training. Although other works have experimented with choosing negative samples from weighted category buckets [1], when sampling non-matches, I assign equal probability to all instances, without weighting by item or category. Examples of positive and negative matches can be found in Figure 2.

3.2.3 Image pre-processing.

I experiment with two methods for pre-processing images. In both methods, I normalize the scale of the garments by cropping the image to the bounding box of the clothing and then resizing the crop to a constant size. I also provide the garment's segmentation mask to the network, to help it distinguish between relevant pixels and background [3].

Preprocessing method 1: masked color. For this method, I crop to the bounding box of the garment and mask out the background with black. This technique pre-

[3]For convenience, I used the ground-truth bounding box and segmentation annotations from DeepFashion2 to perform these preprocessing steps; however, for a purely automated pipeline, one could simply train a semantic segmentation network like Mask R-CNN [10] to predict clothing segmentation masks using the DeepFashion2 data.

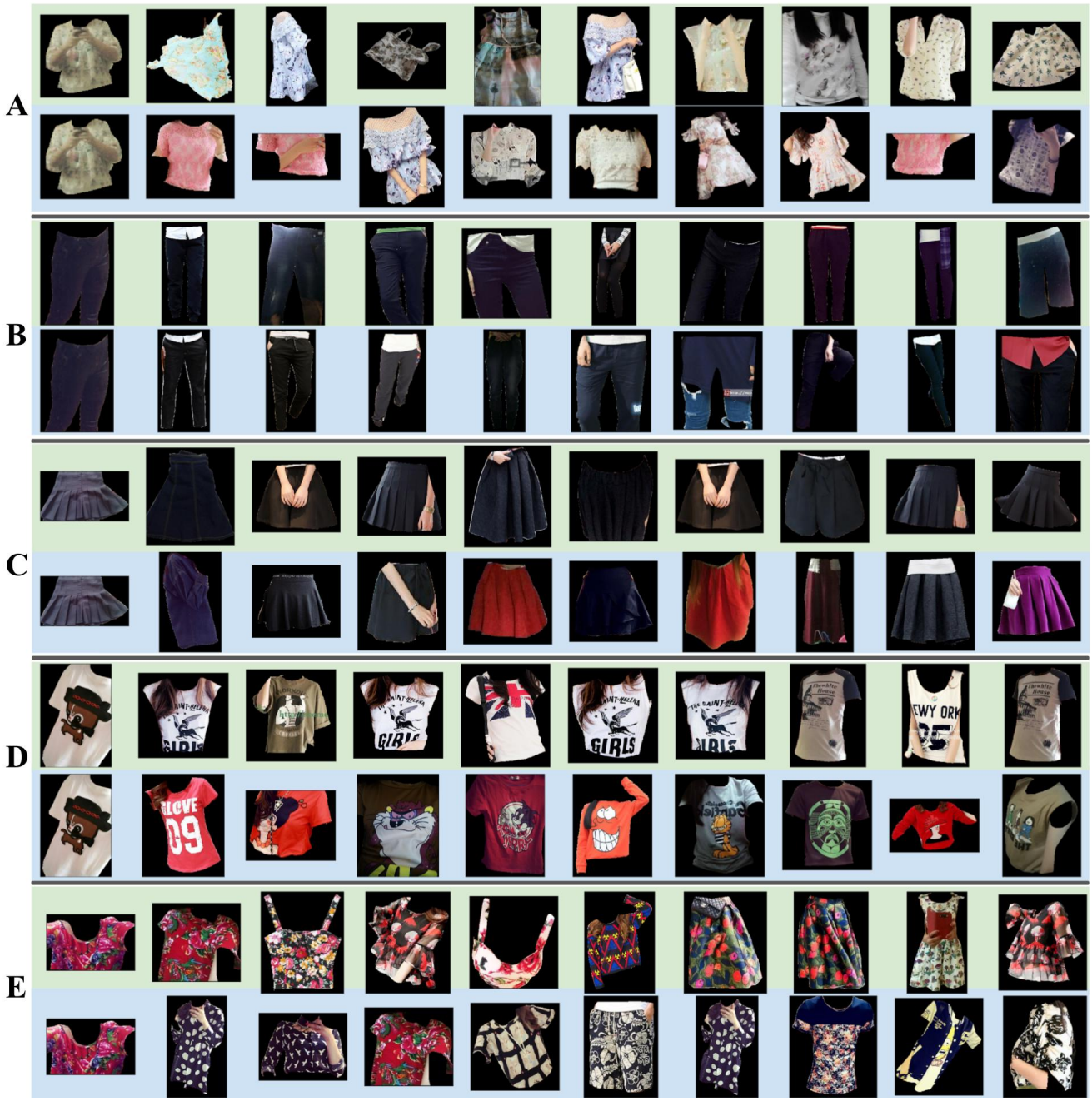

Figure 3. **Example retrieval results from an appearance-based clothing embedding.** The first column is the probe image; the remaining columns are the nearest neighbors in the embedding space. Green rows are from the embedding trained using the masked color preprocessing approach, and blue rows are from the grayscale + mask approach. **Examples A and B** show successes of the encodings, where they produce relevant garments (flowy blouses in Example A, blue jeans in Example B). **Examples C and D** demonstrate a recurring difference between the two encodings. The color model groups items based on color, while the grayscale model is often able to generalize to clothing items with a similar structure (pleated skirts in Example C, graphic tees in Example D) but in a different color family. **Example E** is a failure case where both models prioritize texture over structure, returning floral skirts, dresses, jackets, and shirts (however note that while the color model returns garments in a warm color family, the grayscale model produces busy textures in a variety of colors).

serves full color and texture information from each garment. However, it is less robust to imperfect segmentations since all background pixels that fall outside of the segmentation mask are removed. See Figure 2 (left).

Preprocessing method 2: grayscale + mask. I also tried a grayscale approach to the input data, to see if this would make the network less sensitive to texture information. This involved converting the three-channel color image to a one-channel grayscale image, and then appending the binary segmentation mask as a second channel. Example input images are in Figure 2 (right).

3.3. Results.

To evaluate how effective the learned embedding is, I perform a retrieval task. I pass the images from the DeepFashion2-Duplicates validation set through the model to get their feature vectors. Then, I calculate the 10 nearest neighbors for each image. Some example results are in Figure 3. The encoding is generally succesful, returning relevant results. The grayscale + mask embedding is more likely to return garments with a similar structure and different color than the masked color embedding, but both are still sensitive to textural element such as busy patterns.

4. A structure-based clothing embedding

A common failure mode of clothing embeddings is that they prioritize texture over structure (see Example E of Figure 3, or the sleeveless and long-sleeved shirts returned in response to a T-shirt in Example D). While the combination of grayscale input and a structural segmentation mask (Section 3.2.3) leads to inreased generalizability on the basis of structure, there is still work to be done. In this section, we explore using explicit structural data in the form of simple fashion sketches to learn a texture-invariant embedding.

4.1. The Simplicity patterns dataset.

One domain in which is clothing structure is important is sewing. A sewer can pair an arbitrary design with an arbitrarily-colored fabric, meaning that texture and structure are totally separable features. Thus, sewing patterns provide a rich domain of data for this task. In particular, sewing patterns often come with one or more full-color natural images demonstrating a constructed version of a design, as well as a "line drawing" that better shows off the detail of the design (Figure 4). These photographs and line drawings provide paired data that can be used to train an embedding.

4.1.1 Collecting the dataset.

To gather the necessary data, I scraped images from the Simplicity patterns website. The images associated with a pattern vary in nature and include natural images of models wearing the designs, line drawings, the graphic designs shown on the front and back of the pattern envelope, and colored illustrations of the garment. As such, I had to filter the data to the two former categories. I removed any images with white backgrounds to remove the pattern envelopes and sketches. I then discarded any patterns that did not have a line drawing or at least one natural image.

Figure 4. **Images from a sewing pattern from Simplicity.com.** Left: the pattern envelope marketing the design. Center: a photograph of a garment made using the pattern. Right: a stylized black-and-white "line drawing" showing the design details, stripped of texture information.

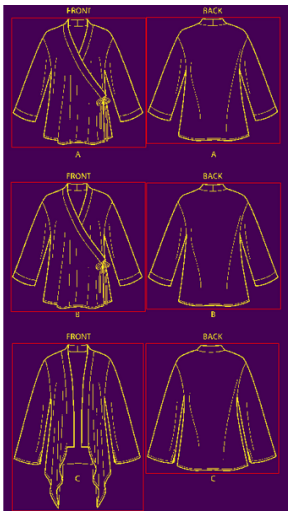

Figure 5. **Automatically-detected crops from a Simplicity line drawing.**

In addition, it is frequently the case that many line drawings for a pattern are combined side-by-side in the same image. Thus, I had to crop out the individual line drawings. I used an out-of-the-box connected component detector to generate bounding box proposals for each sketch in the line drawing and filtered out the proposals that took up less than 1% of the image[4]. An example of the results is in Figure 5.

[4]I also tried using predictions from a pretrained object detection network [18], but it did not generalize well to non-natural images.

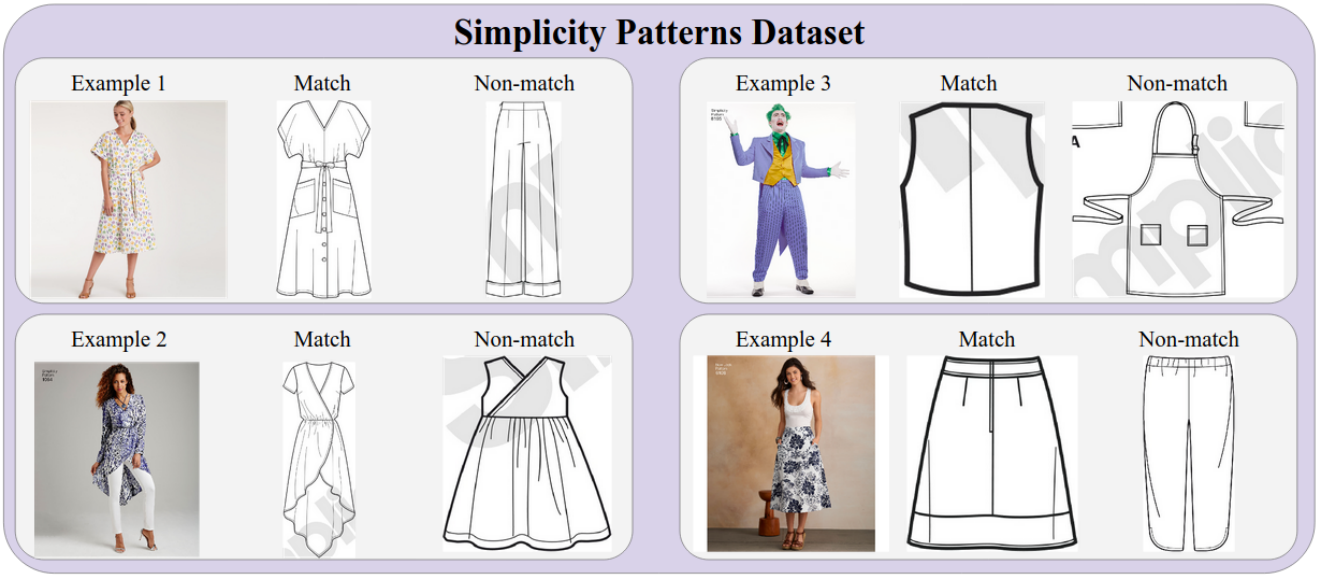

Figure 6. **Example contrastive pairs from the Simplicity patterns dataset.**

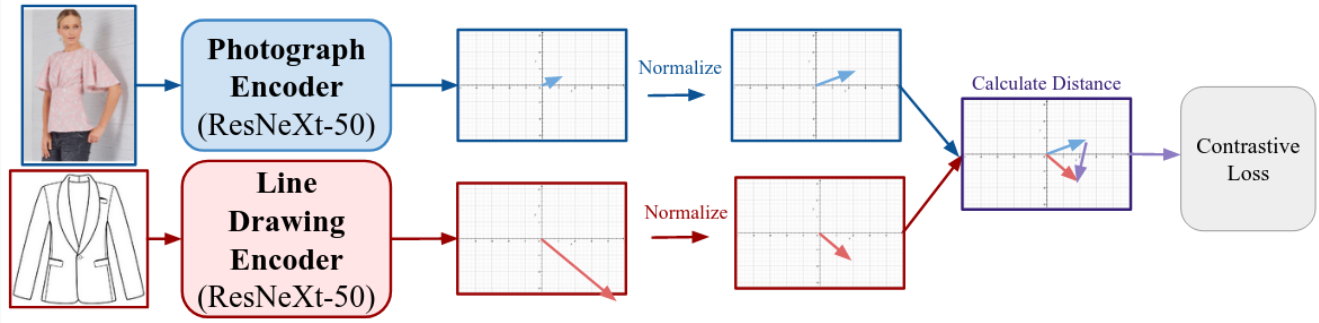

Figure 7. **Architecture of the structure-based clothing embedding model.** Two different encoders with two different sets of weights exist for photographs (blue) versus line drawings (red).

4.1.2 Contrastive loading

The final filtered simplicity dataset contains 1,301 patterns, 4157 images, and 9401 line drawings. This produces over 39 million photograph-line drawing combinations.

As with the Deepfashion2-Duplicates dataset, I construct matching and non-matching pairs of images, except instead of pairing two photographs, I now pair one photograph with one line drawing. During each epoch of training, the model sees one photograph from each pattern in the dataset, randomly selected from the available photographs for that image. With equal probability, the line drawing matches the photograph (in which case it is randomly selected from the line drawing crops for that pattern) or does not match (in which case it is selected from the line drawing crops of all other patterns in the dataset). Example input pairs are shown in Figure 6.

Challenges/limitations. One aspect of this dataset that makes it challenging is the relative lack of human annotations compared to Deepfashion2-Duplicates. At the moment, the dataset does not have semantic segmentations for clothing photographs, so the input to the model is not cropped or masked, leading to much noisier input. Furthermore, when a pattern contains multiple styles of garments, there may be multiple garments featured in the photographs and multiple styles shown in the line drawings (see for example Figure 5). Thus, even among matched pairs, the styles may not perfectly align.

4.2. Model

For this dataset, I tweak the approach from Section 3.1. Instead of using a Siamese network where both images are passed through the same encoder, I use two separate encoders for photographs and line drawings (Figure 7). The rationale behind this is that natural images and black-and-

white sketches are very different domains, and different weights might be required to adequately process these two different image types. The normalization and contrastive loss applied to the encoded vectors remains the same. I initialize both encoders with the weights learned in Section 3. The two encoders will be trained to map a garment's photograph and its structural line drawing to the same point in feature space, which should encourage the network to discard irrelevant texture information.

4.3. Results.

Similar to the analysis in Section 3.3, I use the trained embedding to perform a nearest-neighbors retrieval task. The results are in Figure 8. In general, the results returned by the structural embedding are not as consistent as those returned by the appearance embedding; this could be because the dataset is smaller with less normalization. However, the embedding is much less dependent on color than the appearance embedding, and it sometimes picks up on subtle design elements that might get lost in image-only input (rows A-C of Figure 8) These types of subtle design elements are relevant for a designer or sewer looking to craft a garment, and this shows promise for this technique.

5. PCA of the embedding space

A benefit of the semi-supervised embedding approach is that handcrafted labels such as categories are not required to learn a meaningful embedding. The model should automatically learn which image features are important to identify a garment. In order to gain some insight into what features the model may be using to organize the embedding space, we perform a principal component analysis (PCA) on the embedded vectors. For the Deepfashion2-Duplicates valildation set and the Simplicity patterns dataset, we create a sample matrix with dimensions `number_of_samples` $\times$ `embedding_size`. We perform PCA to get the principal components. Then, to visualize the principal components, we rank the samples based on their value along the component axis and select samples along the axis. The results are in Figure 9.

6. Conclusion and future work

While this work shows some promising results, there is definitely room for improvement towards making a robust structural clothing embedding. For example, the Simplicity patterns dataset is small compared to other datasets in computer vision. A larger dataset could be compiled by turning to other pattern vendors with similar digital content (e.g. Vogue Patterns, McCalls, BurdaStyle, Butterick, etc.) or compilation sites like PatternReview.com that serve as digital pattern archives. The dataset could also greatly benefit from additional cleaning and annotation to reduce noise in the data. In terms of modeling, since natural images and line drawings are quite different domains, techniques used for creating multi-modal embeddings might work well. In addition, attention mechanisms have been shown to be useful for aligning different components of the same input (for example, a word in a caption with a region of an image [26]); here, they could be used to allign different design elements of a garment across the two modalities.

In this project, I made progress towards creating a clothing embedding that captures garment structure and is invariant to texture. I showed that an embedding technique based on Siamese networks and contrastive loss can give reasonable results for clothing. I also showed that simple image transformations like converting to grayscale improve texture generalizability but only to a limited extent. To solve this problem, I proposed creating a joint embedding for natural images and texture-free design sketches. The resulting embedding was able to detect subtle design features across articles of clothing, and with more data could lead to more precise structural embeddings for clothing.

References

[1] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.*, 34(4), July 2015.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.

[3] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[5] Haoye Dong, Xiaodan Liang, Xiaohui Shen, Bowen Wu, Bing-Cheng Chen, and Jian Yin. Fw-gan: Flow-navigated warping gan for video virtual try-on. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[6] Yuying Ge, Ruimao Zhang, Lingyun Wu, Xiaogang Wang, Xiaoou Tang, and Ping Luo. A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images. *CVPR*, 2019.

[7] Xintong Han, Xiaojun Hu, Weilin Huang, and Matthew R. Scott. Clothflow: A flow-based model for clothed person generation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[8] Xintong Han, Zuxuan Wu, Weilin Huang, Matthew R. Scott, and Larry S. Davis. Compatible and diverse fashion image inpainting. *CoRR*, abs/1902.01096, 2019.

[9] Xintong Han, Zuxuan Wu, Zhe Wu, Ruichi Yu, and Larry S. Davis. Viton: An image-based virtual try-on network. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7543–7552, 2018.

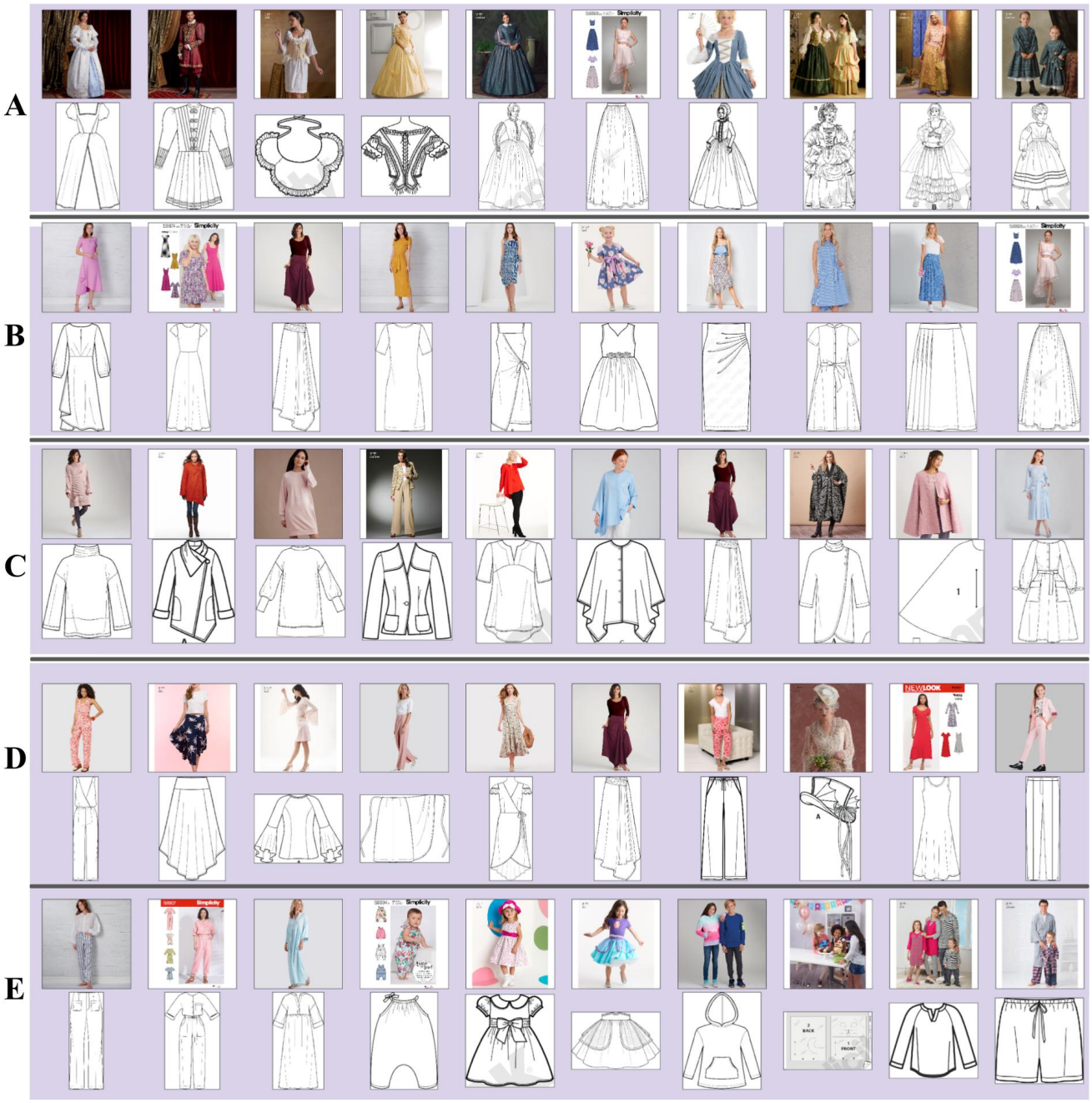

Figure 8. **Example retrieval results from a structure-based clothing embedding.** The first column is the probe image; the remaining columns are the nearest neighbors. All rows use the same embedding space, but the photo rows (top row of each section) were computed using the photo encoder and photographic neighbors, while the line drawing rows used the line drawing encoder and drew from line drawings as neighbor options. However, the model did a really good job of aligning the photograph and line drawing representations, as the two techniques produced the same set of patterns as neighbors! **Examples A, B, and C** show successes for the model, where it picks up on certain structural elements in the garment: full skirts for Example A, skirt pleating/draping (esp. asymmetrical) for Example B, and gathering at the neck for Example C. **Examples D and E** show failure cases where the results returned by the model do not match the input image and/or are very inconsistent. In general, the embedding seems to struggle with pants and jumpsuits, possibly because they are under-represented in the dataset.

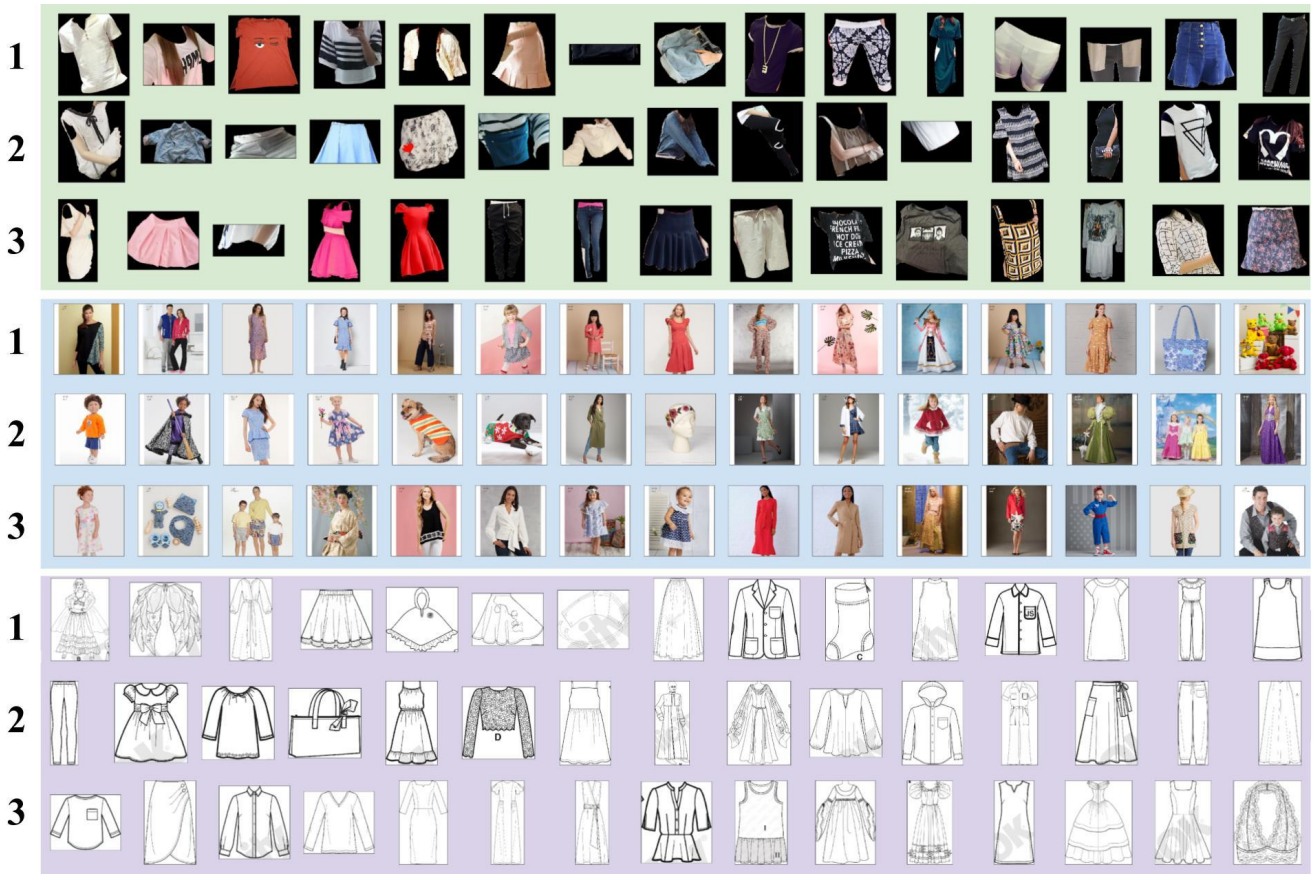

Figure 9. **Principal component visualization.** The green, blue, and purple sections visualize the first three principal components from the gray + mask appearance embedding, structural embedding (photographs), and structural embedding (line drawings), respectively. After calculating the principal components, we rank the samples along the component dimension and visualize 10 samples evenly from the ranking. Note that component 1 from the appearance embedding (green) appears to capture garment type, moving from tops to bottoms. Similarly, component 1 from the structural embedding/photographs section (blue) also seems to correspond to item type, placing non-clothing items like bags and stuffed animals at one end. Component 2 (blue) may also be capture differences in the image background, as the backgrounds become darker and more cluttered from let to right; this is unfortunate because background should not influence the embedding.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[11] Wei-Lin Hsiao and Kristen Grauman. Learning the latent "look": Unsupervised discovery of a style-coherent embedding from fashion images. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4213–4222, 2017.

[12] Wei-Lin Hsiao, Isay Katsman, Chao-Yuan Wu, Devi Parikh, and Kristen Grauman. Fashion++: Minimal edits for outfit improvement. In *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[13] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[14] Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. A generative model of people in clothing. *CoRR*, abs/1705.04098, 2017.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, G.s Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 10 2013.

[16] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[17] Amit Raj, Patsorn Sangkloy, Huiwen Chang, James Hays, Duygu Ceylan, and Jingwan Lu. Swapnet: Image based garment transfer. *European Conference on Computer Vision, ECCV*, 2018.

[18] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[19] Raquel Urtasun Dahua Lin Chen Change Loy Shizhan Zhu, Sanja Fidler. Be your own prada: Fashion synthesis with structural coherence. In *International Conference on Com-*

puter Vision (ICCV), 2017.

[20] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[21] Mariya I. Vasileva, Bryan A. Plummer, Krishna Dusad, Shreya Rajpal, Ranjitha Kumar, and David A. Forsyth. Learning type-aware embeddings for fashion compatibility. *CoRR*, abs/1803.09196, 2018.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[23] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.

[24] Zhirong Wu, Yuanjun Xiong, X Yu Stella, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[26] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015.

[27] L. Yu, E. Simo-Serra, F. Moreno-Noguer, and A. Rubio. Multi-modal embedding for main product detection in fashion. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2236–2242, 2017.